An Optimized Soft Computing Based Passage Retrieval System

Ortiz-Arroyo, Daniel; Christensen, Hans Ulrich

Published in:
Control and Cybernetics

Publication date:
2009

Document Version
Early version, also known as pre-print

Control and Cybernetics

vol. **38** (2009) No. 2

An optimized soft computing-based passage retrieval system*

by

Daniel Ortiz-Arroyo and Hans Ulrich Christensen

Department of Electronic Systems
Aalborg University
Esbjerg Institute of Technology
Niels Bohrs Vej 8, 6700 Denmark
e-mail: do@cs.aaue.dk, hufnc@tdcadsl.dk

Abstract: In this paper we propose and evaluate a soft computing-based passage retrieval system for *Question Answering Systems (QAS)*. *FuzzyPR*, our base-line passage retrieval system, employs a similarity measure that attempts to model accurately the question reformulation intuition. The similarity measure includes fuzzy logic-based models that evaluate efficiently the proximity of question terms and detect term variations occurring within a passage. Our experimental results using *FuzzyPR* on the TREC and CLEF corpora show that our novel passage retrieval system achieves better performance compared to other similar systems. Finally, we describe the performance results of *OptFuzzyPR*, an optimized version of *FuzzyPR*, created by optimizing the values of *FuzzyPR* system parameters using genetic algorithms.

Keywords: information retrieval, question answering systems, passage retrieval, fuzzy logic, optimization, genetic algorithms.

1. Introduction

A *Question Answering System (QAS)* is one type of information retrieval (IR) system that attempts to find exact answers to user's questions expressed in natural language. In an *Open-Domain Question Answering System (ODQAS)*, questions are not restricted to specific domains and their answers are commonly searched for within an unstructured document collection. Building effective ODQAS for unstructured document collections is a challenging task due to the complexity associated with processing natural language.

QAS are typically constructed in a pipeline fashion. First, a question analyzer module identifies the type of question submitted by a user. Afterwards, a document retrieval system retrieves a group of documents that are relevant

*Submitted: December 2007; Accepted: July 2008.

to the type of posed query. Then, a *Passage Retrieval (PR) system* extracts text segments from the group of retrieved documents and ranks these passages in decreasing order of computed likelihood for containing the correct answer to a question. Typically, such text segments are referred to as *candidate passages*. Finally, the closest answer to a question is extracted from the set of passages and presented to the user.

Due to its pipelining structure, the overall performance of a QAS is limited by the performance achieved by each of its components. One of the most critical components of a QAS is the PR system. A PR system that fails to retrieve any answering passages to a question, or that returns many, large candidate passages, will have a negative impact on the overall effectiveness of the QAS (Gaizauskas et al., 2003).

Previous research (Brill et al., 2001; Gómez-Soriano et al., 2005a) has proposed to use the *question reformulation* intuition as an effective model to implement QAS, concretely within its PR component. The question reformulation intuition can be expressed informally as: "*frequently, an answer to a (factoid) question is found as a reformulation of the same question*".

An example of applying the *reformulation intuition* in a QAS is looking for the answer to the question: "How much is the international space station expected to cost?" of QA@TREC 11 (QID:1645)[1]. The answering passage to that question contains the snippet: "(...)United States and Russia, are working together to build the SPACE STATION, which is EXPECTED TO COST between $40 billion and $60 billion.(...)".

The previous answering passage illustrates that one simple way to find the answer to the question posed, is to look for passages containing *most* of the same terms employed in formulating the question or *variations* of these terms. Additionally, the example also shows that generally the question terms are located in *close proximity* within the answering passage. This simple example shows that a straightforward way to design a PR system is to include fuzzy logic-based models capable of handling the vague concepts employed in the reformulation intuition such as *most*, *variations*, and *close proximity* using fuzzy sets.

Following this idea, this paper presents a novel passage retrieval system touted *FuzzyPR* and its optimized version *OptFuzzyPR*. *FuzzyPR* is a language-independent PR system for ODQAS that has the following additional features: a) it is based on a soft computing approach, b) it handles factoid questions, c) it provides a novel implementation of the reformulation intuition using a similarity measure, and d) it was especially tuned to optimize its performance. The soft computing approach we have used employs a fuzzy logic-based similarity measure aimed at evaluating the similarity between the retrieved passages and the question posed, in terms of the reformulation intuition. The similarity measure was built by modeling the proximity of question terms, the term variations oc-

[1]TREC's Question Answering collections are available from:
`http://trec.nist.gov/data/qa.html`

curring within a passage, and using arbitrary passages of a fixed length, through fuzzy sets. The models included in the similarity measure were selected empirically, as they showed the best performance in our experiments. Finally, we describe *OptFuzzyPR*, which is an optimization of the baseline *FuzzyPR* system obtained by the use of Genetic Algorithms (GA).

OptFuzzyPR has been implemented within a full QAS, especially built to assess the performance of our PR system.

The paper is organized as follows. Section 2 describes previous related work on passage retrieval systems for QAS. Section 3 describes and analyzes in detail the main component mechanisms of *FuzzyPR*, our passage retrieval system. Section 4 describes *FuzzyPR* and its performance results. In Section 5 we describe an optimization method based on Genetic Algorithms, that we used to create *OptFuzzyPR*, and compare its performance results with those obtained by the non-optimized version. Finally, Section 6 presents some conclusions and future work.

2. Related work

Recent research work on passage retrieval systems for QAS may be roughly classified into methods based mostly on the application of natural language processing (NLP) techniques and those based on lexical and/or statistical information extracted from questions and corpora.

Among the methods based on NLP, two recent studies, Tiedemann (2005) and Cui et al. (2005) explore the application of diverse NLP-based techniques in passage retrieval systems. Tiedemann (2005) shows an improvement in the Mean Reciprocal Rank (MRR)[2] of 19% using an optimized passage retrieval system when compared to an un-optimized baseline system. His passage retrieval system employs automatic learning of feature selection and query optimization using genetic algorithms on the test set. This approach employs information generated by performing a deep syntactic analysis of passages and questions.

Cui et al. (2005), propose a ranking method based on the approximate (fuzzy) comparison of syntactic dependencies of questions and passages (sentences). By taking the grammatical structure into account they intend to minimize the number of false positive passages obtained, compared with the application of methods based on term density. A parser touted Minipar, is used for extracting the grammatical structure. To make a fuzzy comparison between question and sentence structures, the IBM translation model is employed (Brown et al., 1993). The authors implemented this method in three existing PR systems (MITRE, SiteQ and NUS) and tested it on the TREC's QA corpus called ACQUAINT. They found that the proposed method achieves a statistically significantly improved MRR and precision of the top passages, ranging as high as 0.4924, which is a 83.93% improvement over the baseline.

[2]MRR is defined in Section 3.1.

Saggion et al. (2004) and Unsunier, Amini and Gallinari (2004) propose methods based on linguistic resources. Saggion et al. (2004) use WordNet for term expansion, and Unsunier, Amini and Gallinari (2004) use the same WordNet but for the generation of ranking features. Both approaches propose retrieval strategies that include techniques such as term (synonym) expansion, dynamic matching windows, and deletion of query terms in order to broaden queries. They also look at query formulation and reformulation, as the use of strict Boolean AND and OR operators may cause too few or too many documents to be retrieved.

Saggion et al. (2004) found that while their best performing retrieval strategy, called StrLteMorph20, achieves Coverage[3] of 62.15% at rank 200, the standard PR system Z-PRISE performs better by achieving Coverage of 80.4%. However, on average the Boolean strategy retrieves only 137 sentences per question, which is preferable for answer extraction, whereas Z-PRISE returns around 4600.

The following methods rely on extracting lexical or statistical information from passages. These methods are closer in scope to the one applied in *FuzzyPR*.

Unsunier, Amini and Gallinari (2004) explore the application of the RankBoost algorithm in passage retrieval with the purpose of improving both coverage and precision. RankBoost is a machine learning method that combines the results of weak learners (in this case binary decision functions) to rank features. The method was originally introduced by Freund et al. (2003) in the domain of collaborative filtering. Passages are first associated with local scores that measure the relevance of the passage to answering the question. Then these scoring functions are used during training by RankBoost. An adapted version of the RankBoost algorithm presented in Unsunier, Amini and Gallinari (2004), achieves consistently higher coverage, reaching 72.8% at rank 20, when compared to three other methods: a standard passage retrieval system, the MG (Managing Gigabytes) search engine[4] and a linear support vector machine. One disadvantage of this approach is that it requires a training set. In the experiments reported, approximately 150 feature objects were used for training and 150 questions were used for testing.

Terra and Clarke (2005) investigated different query formulation and query expansion strategies using related terms, extracted statistically from a large corpus. They found that query expansion does not result in the expected improvement on precision when compared to query formulation. The authors hypothesize that a combination of both techniques might yield a better performance.

Huang, Huang and Wu (2004), propose an improvement of IBM's BM25 passage retrieval algorithm (Robertson et al., 1995). In this approach the terms of the surroundings of a matching term, within a defined window size, are given

[3]Coverage is defined in Section 3.1.

[4]Home page of the MG book and search engine: `http://www.cs.mu.oz.au/mg/`

weights according to their distance from the matching term. This method is called HotSpot. Two additional measures are also taken into account when the question-passage similarity is calculated: Height, which is the maximum weighted term of HotSpot, and Coverage, which is the fraction of distinct query terms of the passage. An experiment carried out with ACQUAINT showed that their proposed blurred variation of the BM25 algorithm improved both Coverage (18.3%) and Answer Redundancy[5] (4.8%).

Monz (2004) introduces and evaluates a novel proximity-based weighting method for document retrieval called *Minimal Span Weighting (MSW)*. The idea of the method is based on the notion of a minimum matching span, which he defines as the smallest document fragment containing all the words of the query. The length of the minimum matching span is used for re-calculating the *retrieval status value* (RSV) of a matching document during document retrieval. MSW achieves a statistically significant improvement in terms of Mean Reciprocal Rank (MRR) when tested with the data sets of TREC 9, 10 and 11.

Gómez-Soriano et al. (2005b) introduce a novel n-gram based PR method called JIRS, that was adapted to the special needs of QA. JIRS is an n-gram based method that re-ranks passages retrieved by a vector space model-based passage retrieval system, giving the highest RSV to those passages containing the longest sequence of matching terms from the question. JIRS supports two extensions to the basic n-gram matching mechanism (called *Simple Model*): term weights model (called *Term Weight*) and both term weights and a distance measure model (called *Distance Model*). In summary, JIRS basically ranks higher passages containing larger sequences of the terms contained in the questions. In Gómez-Soriano et al. (2005b) it is shown that JIRS is capable of outperforming other similar methods.

In order to handle syntactic variations between questions and passages, Vilares and Alonso (2004) propose the application of locality-based retrieval for re-ranking, a method originally introduced by de Kretser and Moffat (1999a,b). However, the model was adapted to passage-based retrieval[6] A comparative evaluation of this technique with SMART, a vector space model engine that uses a weighting scheme, shows that effectiveness is not improved. Further analysis of the results reveals that fusing the results produced by the SMART document retrieval method and the locality-based methods may be beneficial. The application of data fusion shows an improved precision for top ranked documents for both short and long queries.

Kong et al. (2004) use fuzzy aggregation operators in a passage-based retrieval system for documents, where the relevance of a document is re-calculated taking into account the retrieved passages.

As the short summary of previous related research work on passage retrieval

[5]Redundancy is defined as the average number, per question, of the top n passages, which contain a correct answer.

[6]Essentially, passage-based retrieval is document retrieval, where the occurrence of matching passages is taken into account to calculate a similarity score (Kong et al., 2004).

systems for QAS reveals, numerous approaches have been proposed to improve the performance of the PR system. Some of these approaches have explored the use of concepts similar to those applied in the *reformulation intuition*, as in Brill et al. (2001) and Gómez-Soriano et al. (2005a).

The approach presented in this paper differs from the previous works in using an optimized soft computing-based approach to create a question-passage similarity measure aimed at modeling more accurately the reformulation intuition.

3. Analysis of main components of a passage retrieval system

As was previously indicated, the *reformulation intuition* can be modeled using two characteristics of a candidate passage: "*most (important) question terms*" occurring within a passage in "*close proximity*". The feature *"most (important) question terms"* can be modeled by the fuzzy subset: *The degree to which candidate passages contain m out of n question terms.* The degree of membership varies from 1, when all important question terms occur within a candidate passage, to 0, if no question terms occur within the passage.

"Close proximity", on the other hand, can be modeled by the fuzzy subset: *The degree to which the question terms contained in a candidate passage are adjacent.* If all question terms of the passage are adjacent, then the passage membership degree in this fuzzy subset is 1. Otherwise, the more distributed the terms are in a passage, the lower the degree of proximity, approaching 0. It must be remarked that we use the term proximity to refer to the physical distance of the terms within the passage.

When the terms employed in the question and the answering passage have exactly the same form, the two features *"most important question terms"* and *"close proximity"* may be enough to implement the reformulation intuition. However, questions and documents commonly contain grammatical inflections and typos that, if not handled adequately, will have a negative impact on QAS performance. Therefore, to cope with this situation, the third vague concept that can be used in the reformulation intuition is *"'term similarity (matching)"*.

The fuzzy logic interpretation of term similarity is the fuzzy subset: *The degree to which two terms are similar*, yielding 1, if the two terms are identical, a value between $]0, 1[$ if they have some letters in common, and 0, if they are very different.

In the following subsections we briefly describe and analyze some fuzzy logic-based models that can be used to implement the reformulation intuition, applying the concepts of *proximity of question terms occurring in a passage* and *term similarity*. The implementation, using fuzzy logic, of the concept "*most (important) question terms*" is described in Section 4.2.

3.1. Proximity of question terms occurring in a passage

Fuzzy proximity measures calculate the proximity degree (in terms of physical distance) of two or more question terms contained within a passage in a document. The measure implements the following model: 1) if all question terms are adjacent in a document then the measure yields 1, and 2) the farther away the terms occur in the document, the lower the degree of proximity.

We evaluated three different fuzzy proximity measures, as to their ability in finding answering passages for the first 50 questions of TREC11 question set using the ACQUAINT corpus. It should be noted that we have extended two of the measures using fuzzy sets with the idea of improving its performance. During the evaluation we used the standard QAS evaluation metrics *Mean Reciprocal Rank (MRR)* and *coverage*. *MRR* is defined as the average of the reciprocal rank r_i of the first hit to each question within the top five candidate passages:

$$MRR = \frac{1}{|Q|}\sum_{i=1}^{|Q|} RR_i \tag{1}$$

where $RR_i = \frac{1}{r_i}$ if $r_i \leq 5$ or 0 otherwise and Q is the set of questions. As it is done in the JIRS system (Gómez-Soriano et al., 2005a), we measured coverage on the first top 20 passages[7]. *Coverage* is defined as the proportion of questions for which an answer can be found within the n top-ranked passages:

$$cov(Q, D, n) \equiv \frac{|\{q \in Q | R_{D,q,n} \cap A_{D,q} \neq \emptyset\}|}{|Q|} \tag{2}$$

where Q is the set of questions, D is the passage collection, $A_{D,q}$ the subset of D containing correct answers for $q \in Q$ and $R_{D,q,n}$ the n top ranked passages.

The first proximity measure employed in our experiments called *Fuzzy Proximity* measure, was proposed by Beigbeder and Mercier (2005). The key point of their method is the modeling of *relative position* of terms by means of a proximity function. Essentially, the proximity function is a fuzzified version of the NEAR operator used in Boolean Information Retrieval models. Although the proximity function may be as complex as a Gaussian function, Beigbeder and Mercier (2005) report achieving good result with a simple triangular function given by (3):

$$\mu_t^d(x) = \max_{i \in Occ(t,d)} \left(\max \left(\frac{k - |x - i|}{k}, 0 \right) \right) \tag{3}$$

where x is the position of a term in the document, $Occ(t, d)$ is the set of positions where the term t is occurring in the document d, and the constant k determines the influence of a term — i.e. the support of the Fuzzy Subset. The function (3) has two important properties:

[7] Also called coverage@20 for short

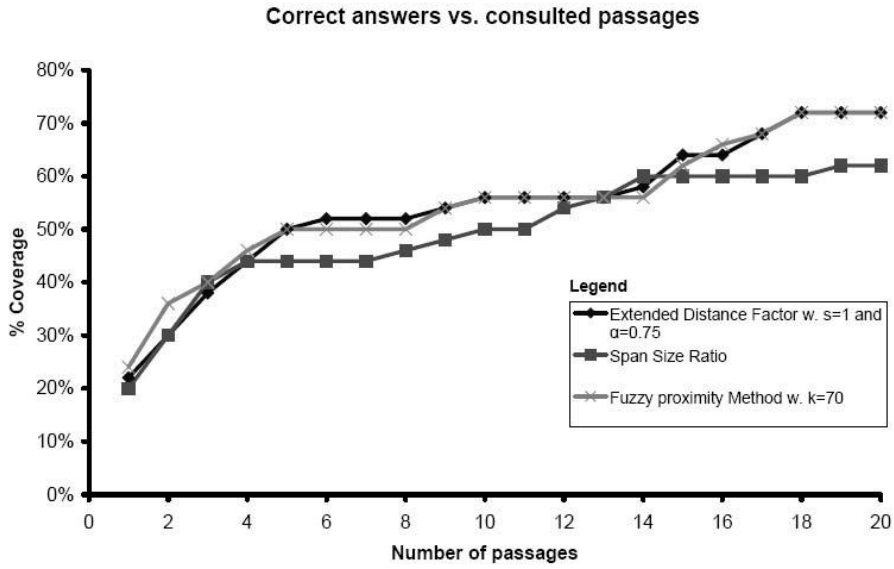

Figure 1. The Coverage of the best performing runs for each fuzzy keyword proximity measure.

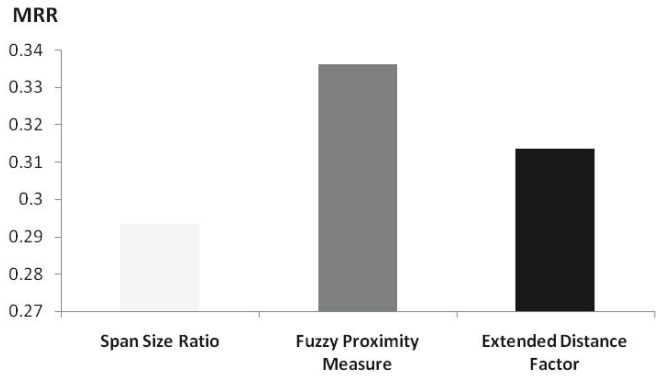

Figure 2. The MRR of the best performing runs for each fuzzy keyword proximity measure.

- The outer *max* resolves the situation, in which a term occurs multiple times in a document: The one closest to the center, i.e. having the *highest* degree of proximity, is chosen.
- The inner max ensures that $\mu_t^d(x) \in [0, 1]$.

The second proximity measure called *Extended Distance Factor* is an improvement we have made over JIRS PR4QA (Gómez-Soriano et al., 2005b) system weighting scheme, in such a way that it prefers *heavy*[8] rather than *long* n-grams. Gómez-Soriano et al. (2005b) introduce and apply the concept of *Distance Factor*, whose formula is:

$$dist(x, x_{max}) = \frac{1}{1 + k * ln(1 + D(x, x_{max}))} \tag{4}$$

where:

- x and x_{max} are n-grams, with x_{max} being the n-gram with the maximum weight defined by $w_k = 1 - \frac{log(n_k)}{1+log(N)}$, n_k being the number of passages where term t_k appears, and N the number of passages retrieved;
- $D(x, x_{max}) = |a - b|$, where a and b are the locations of the n-grams x, x_{max}, respectively, in the passage, and;
- $k \in R_+/\{0\}$ is a parameter adjusting the importance of the distance. If $k \in]0, 1[$ the importance of the weights attached to distant terms is increased, and if $k \in]1, \infty[$ the importance of the weights attached to distant terms is decreased.

In the Distance Factor n-gram based model a passage containing more keywords gets a higher weight despite not containing the longest n-gram. A closer look at the Distance Factor reveals that in fact it is a locality-based normalized proximity measure resembling the membership function of the binary fuzzy subset $\mu_{CPoK}(t_1, t_2)$: *Close proximity of two terms* as shown in (5). Thus, besides n-grams it is also applicable to terms in general.

$$\mu_{CPoK}^{DM}(t_1, t_2) = \frac{1}{1 + s * ln(1 + dist(t_1, t_2))} \tag{5}$$

However, since distance of Gómez-Soriano et al. (2005b) is between two n-grams, it is necessary to extend it so that it measures the proximity of n keywords, as required. One way of doing this using fuzzy logic is by taking the AND-like average[9] of the proximity of all binary permutations of matching keywords as in (6):

$$\forall t_i, t_j \in P \cap Q(t_i \neq t_j), h_\alpha \left\{ \mu_{CPoK}^{DM}(t_i, t_j) \right\}. \tag{6}$$

Equation (6) is an extension to the original JIRS distance measure that we introduce in this paper and that we called *Extended Distance Factor*, where $\alpha \in]0.5, 1.0]$ is the degree of ANDness and h_α an Averaging Operator.

[8] We used *heavy* to mean n-grams containing more question terms

[9] By taking the AND-like average we relax the criterion of close proximity.

Finally, the last approach we used called *Span Size Ratio* (SSR) is a fuzzification we have made of a method originally introduced by Monz (2004). SSR is defined in Monz (2004) as *"the number of unique matching terms in the span over the total number of tokens in the span"*. Basically, SSR may be seen as a density-based retrieval algorithm resembling the fuzzy subset of *close keyword proximity*, given in (7):

$$\mu(q,d)_{CPoK}^{SpanSizeRatio} = \frac{|q \cap d|}{1 + max(ms) - min(ms)} \tag{7}$$

where a *matching span* (ms) is a set of positions that contains at least one position of each matching term and $max(ms)$ and $min(ms)$ are the positions of the first and the last matching keywords of the matching span, respectively.

We implemented the three proximity measures previously discussed and evaluated their performance. Figs. 1[10], 2 show that *Fuzzy Proximity Measure* of Beigbeder and Mercier (2005) achieves the same level of coverage at ranks 1-20 as the *Extended Distance Factor*, but performs 7.2% better in terms of MRR.

3.2. Term similarity

Term variations are lexical differences — in terms of meaning and spelling — between a word of the question typed by a user and an equivalent word contained in a document in the corpus. Reasons for the occurrence of term variations in natural language include grammatical inflection and spelling mistakes. Two main features are needed in a mechanism to handle term variations effectively: 1) *language-independence* and 2) *effectiveness*, measured as the tolerance toward common misspellings and grammatical inflections. Term similarity algorithms handle term variations efficiently.

Several algorithms have been proposed in the literature (Levenshtein, 1966; Damerau, 1964; Szczepaniak and Gil, 2003) to determine the degree of similarity between two strings. The degree of similarity may be defined as the inverse of the degree of dissimilarity.

$$\mu_{sim}(t_1,t_2) = 1 - \mu_{dissim}(t_1,t_2). \tag{8}$$

The *Longest Common Subsequence* (LCS2) algorithm, a classical method employed in computer science, can be used to construct a similarity measure. Contrarily to a substring, a subsequence needs not to be continuous and non-matching characters may be interleaved. As an example, the longest common subsequence between "Etymology" and "Etymlogeys" is "Etymlogy". As the longest common subsequence measures the commonality between two strings, a term similarity measure can readily be constructed by normalizing it by the length of the longest term:

$$\mu_{termsim}^{LCS2} = \frac{LCS2(t_1,t_2)}{max(|t_1|,|t_2|)}. \tag{9}$$

[10]The parameters shown in the figure are described in Beigbeder and Mercier (2005).

This approach is identified in Table 1 as *Normalized longest common subsequence*.

Edit Distance is defined as the minimum number of edit operations necessary to transform a term t_1 into another term t_2. Two prevalent Edit Distance algorithms are the *Levenshtein Distance* (LD) (Levenshtein, 1966) and the *Damerau-Levenshtein Distance* (DD) (Damerau, 1964). The Levenshtein Distance defines edit operations as insertions, deletions, and substitutions. The Damerau-Levenshtein Distance is a variation of the LD with the additional operation of *transposition*. An example on the difference between applying the LD and DD algorithms is calculating the distance between the terms "approixmate" and "approximate". Where LD requires two substitutions, DD only one.

Taking the inverse of the *max* normalized LD and DD, respectively, provides a simple fuzzy string similarity measure. The normalization is based on the fact that $LD(t_1, t_2) \leq max(|t_1|, |t_2|)$. That is, at most $max(|t_1|, |t_2|)$ LD edit operations are necessary in order to transform t_1 into t_2. Therefore, $\frac{LD(t_1,t_2)}{max(|t_1|,|t_2|)} \in [0, 1]$. Noting that the normalized LD and DD both are the membership functions of the fuzzy subset (relation) of dissimilar terms, by applying (8) we get the fuzzy similarity algorithms of (10) and (11) below. These algorithms are identified in Table 1 as *Inverse Normalized DD and LD*, respectively:

$$\mu_{sim}^{inDD}(t_1, t_2) = 1 - \mu_{dissim}(t_1, t_2) = 1 - \frac{DD(t_1, t_2)}{max(|t_1|, |t_2|)} \tag{10}$$

$$\mu_{sim}^{inLD}(t_1, t_2) = 1 - \mu_{dissim}(t_1, t_2) = 1 - \frac{LD(t_1, t_2)}{max(|t_1|, |t_2|)}. \tag{11}$$

Szczepaniak and Gil (2003) introduced an n-gram based fuzzy term similarity matching algorithm, based on the exhaustive comparison of all possible substrings ranging from *unigrams* to N-grams, where $N = max(|t_1|, |t_2)$. The measure is shown in (12) and (13):

$$\mu_{sim}^{sg}(t_1, t_2) = \frac{2}{N^2 + N} \sum_{i=1}^{|t_1|} \sum_{j=1}^{|t_1|-i+1} h(i, j) \tag{12}$$

$$h(i, j) = \begin{cases} 1 & \text{if } substring(t_1, j, i) \in t_2 \\ 0 & otherwise \end{cases}; \tag{13}$$

$substring(t_1, j, i) \in t_2$ means that a substring containing i characters and beginning from the jth position in t_1 appears in t_2. It must be noted that the multiplication by the factor $\frac{2}{N^2+N}$ is necessary in order to normalize the measure to a value in the unit interval since there are $\frac{N^2+N}{2}$ n-grams (or substrings) to be considered. The result of using previous equations is shown in Table 1 as *Szczepaniak and Gil*.

Lin (1998) noted that similarity measures can be constructed based on distance metrics by taking the reciprocal value of the distance plus 1. Since both

Table 1. Average similarity scores of six fuzzy similarity algorithms (sorted in decreasing order)

Algorithm	**Average similarity score**
Normalized longest common subsequence	0.5984
Inverse normalized DD	0.5569
Inverse normalized LD	0.5513
Szczepaniak and Gil	0.4395
Reciprocal DD	0.3751
Reciprocal LD	0.3720

the LD and DD yield a value between 0 and $max(|t_1|, |t_2|)$, it is possible to apply the transformation directly within the fuzzy term similarity algorithms of (15) and (14).

$$\mu_{sim}^{rDD}(t_1, t_2) = \frac{1}{1 + dist(t_1, t_2)} = \frac{1}{1 + DD(t_1, t_2)} \tag{14}$$

$$\mu_{sim}^{rLD}(t_1, t_2) = \frac{1}{1 + dist(t_1, t_2)} = \frac{1}{1 + LD(t_1, t_2)}. \tag{15}$$

These equations were used to calculate the similarity scores identified as *Reciprocal DD and LD* in Table 1.

We implemented the six term similarity algorithms already described and performed a comparative evaluation of their effectiveness when set to calculate the similarity between 300 English homophone[11] pairs. The average of the similarity computations yields the score of the fuzzy term matching algorithm.

Table 1 shows that the *Normalized longest common subsequence* (nLCS) performed best, giving an average homophone pair similarity rate of 0.5984.

4. *FuzzyPR's* components

FuzzyPR consists of two main components: 1) a question–passage similarity measure component whose model implements the concepts of *proximity of question terms occurring in a passage* and *term similarity*, and 2) a passage identification and extraction component that implements the "*most (important) question terms*" concept within the reformulation intuition. The following subsections describe the similarity measure and the passage identification and extraction algorithm.

[11] A *homophone pair* is two terms pronounced the same but differing in meaning and spelling, thus reflecting misspellings and typos. Examples include "advice vs. advise" and "cite vs. site".

4.1. Similarity measure

The similarity measure we propose is a fuzzy logic-based implementation of the *reformulation intuition*, which can be stated in a more formal way as follows: "*A passage p is relevant to the user's question q if many question terms or variations of these question terms occur in close proximity*". The similarity measure is described by:

$$\mu_{rel}(p,q) = wMin\left((v_1, \mu_f(p,q)), (v_2, \mu_p(p,q))\right) \tag{16}$$

where $wMin$ is the weighted minimum of the two measures $\mu_f(p,q)$ and $\mu_p(p,q)$ and v_1, v_2 are importance weights. It must be remarked that we use the term similarity within the context of the matching degree between a passage and a query.

The similarity measure combines lexical and statistical data extracted at *term-level* into the two fuzzy measures: $\mu_f(p,q)$, the weighted fraction of question terms q occurring in the passage p, and $\mu_p(p,q)$, the proximity of question terms q within the passage. Using the results of the performance analysis described in Section 3, $\mu_f(p,q)$ and $\mu_p(p,q)$ are defined in (17) and (18):

$$\mu_f(p,q) = h_{\alpha_f}\left((v_1^f, sat(p,t_{q_1}))\ldots(v_n^f, sat(p,t_{q_n}))\right) \tag{17}$$

where h is the *importance weighted averaging* (AIWA) operator proposed by Larsen (2003) with an ANDness of $\alpha_f = 0.65$, t_{q_i} is a question term, $v_i^f = NIDF(t_{q_i}) = 1 - \frac{log(n_i)}{1+log(N)}$ [12], n=frequency of t_{q_i} in Ω, the set of documents, $N = |\Omega|$. $sat(p,t_{q_i})$ measures the degree to which p contains t_{q_i} using the normalized longest common subsequence (nLCS), i.e. $sat(p,t_{q_i}) = \max_{\forall t_p \in p}\left(\mu_{sim}^{nLCS}(t_p,t_{q_i})\right)$, where $\mu_{sim}^{nLCS}(t_p,t_{q_i}) = \frac{|LCS(t_p,t_{q_i})|}{\max(|t_p|,|t_{q_i}|)}$, LCS being the longest common subsequence. Finally,

$$\mu_p(p,q) = \frac{s(p,q)}{\max\limits_{p_i \in \Omega} s(p_i,q)} \tag{18}$$

where $\mu_p(p,q)$ is a max-normalization of *fuzzy proximity* method of Beigbeder and Mercier (2005) described by $s(p,q) = \int_1^n \mu_t^p(x)dx$, $t \in q$, with the term influence function $\mu_t^p(x) = \max\limits_{i \in Occ(t,p)}\left(\max\left(\frac{k-|x-i|}{k}, 0\right)\right)$, where the parameter adjusting the support is $k = 70$. The values of v_1, v_2, α_f and k were determined experimentally. Aggregating these two fuzzy measures using the weighted minimum gives the overall relevance score $wMin$, which is defined as:

$$wMin(v_1,v_2,\mu_f,\mu_p) = \min\left(\max(1-v_1,\mu_f(p,q)), \max(1-v_2,\mu_p(p,q))\right) \tag{19}$$

[12] NIDF is an abbreviation of normalized inverse document frequency.

with the importance weights $v_1 = 1$, $v_2 = 1$; μ_f and μ_p evaluated for (p,q) where both the passage p and the question q are represented as sets of terms: $\{t_{p_1}, t_{p_2}, ..., t_{p_n}\}$ and $\{t_{q_1}, t_{q_2}, ..., t_{q_m}\}$, respectively. $wMin$ aggregates $\mu_f(p,q)$ and $\mu_p(p,q)$ into a single fuzzy value $\mu_{rel}(p,q)$ as described by equation (16). $\mu_{rel}(p,q)$ has the additional advantage of being *language-independent*.

4.2. Passage identification and extraction mechanism

To model the vague concept "*most terms*", contained in the reformulation intuition, we used a fuzzified variation of the concept of *fixed length arbitrary passages*[13].

An arbitrary passage is modeled as its membership function in the ideal set of passage sizes as stated in equation (20):

$$\mu_{Ideal\ passage\ size}(x) = \begin{cases} 1 & \text{if } 0 \leq x \leq d \\ \frac{x-b}{d-b} & \text{if } d < x < b \\ 0 & \text{if } x \geq b \end{cases} \tag{20}$$

where x is the term location in the passages and d and b adjust the crisp support and the fuzzy support, respectively.

Due to efficiency concerns, a passage or matching span is defined as a window determined by the symmetric membership function of the ideal passage size around an important, matching term t_c with a similarity to a question term greater than some threshold value α and a NIDF greater than β. This restriction is justified by the intuition that a passage containing none or very few of the question terms is unlikely to provide an answer to the question posed. Since a document term degree of membership in the corresponding symmetric passage counts as a mandatory weight[14], terms with a membership degree of 0 do not belong to the matching span.

Because overlapping passages impose an additional computational burden, these are removed using some simple yet effective rules where those passages with a) higher similarity and b) smaller text span are preferred.

Finally, passages are aligned to having an equal size by incrementally removing less important document terms from them until the ideal passage size is achieved.

5. *FuzzyPR's* performance results

We measured the effectiveness of *FuzzyPR*, the non-optimized version of our passage retrieval system, comparing its ability to find correct answers to questions in a document corpora with *LucenePR*, a PR system that we have integrated

[13] Arbitrary passages are defined as: *"any sequence of words of any length starting at any word in the document"*.

[14] This weight is only used for calculating the fuzzy proximity of question terms.

Table 2. The average passage sizes of the PR systems used in the experiments.

PR system Test data	**TREC12**	**CLEF04**
LucenePR	55.91	74.74
JIRS *Distance* Model	132.23	105.87
JIRS *Simple* Model	166.96	111.48
Arithmetic mean	118.37 (119)	97.36 (98)

within Lucene—a popular vector space search engine—and the JIRS PR system described in Gómez-Soriano et al. (2005a). To create *LucenePR* we implemented a special indexing method in Lucene that segments a document into three sentence passages with one sentence overlapping. In order to process a question, *LucenePR* creates a number of conjunctive queries consisting of all n up to just two non-interrogative question terms. Queries are posed in sequence until at least one answering passage is returned.

Regarding the JIRS PR system, we decided to use in our evaluation the *simple* model and the *distance* model of JIRS (described in Section 3), as we found that the other models included in JIRS, such as the so called *term weighted* model, perform almost identically to the *simple* model.

Documents are divided into passages consisting of three consecutive sentences with one sentence overlapping prior to indexing. This approach was used since Llopis, Ferrández and Luis Vicedo (2002) reported that it generally achieves good results. LucenePR employs a simple query expansion method consisting in removing a term in the question with the lowest IDF until ≥ 20 passages are retrieved from the index of three sentence passages.

To determine *FuzzyPR* ideal passage size, we computed and used the arithmetic mean of the average passage sizes of the top 100 passages retrieved by LucenePR, JIRS *distance* Model and JIRS *simple* Model. In Table 2 the numbers in parenthesis are the actual passage sizes used by *FuzzyPR*.

As test data we used TREC12 set of 413 questions and the corpus called ACQUAINT consisting of $1,033,461$ documents of English news text, and CLEF04 180 question and the AgenciaEFE corpus of $454,045$ Spanish newswire documents. To answer questions automatically for TREC12 we used Ken Litkowsky's regular expression patterns of correct answers[15] and for CLEF4 we used the patterns supplied with JIRS[16]

The original TREC12 question set was reduced to 380, since 115 questions do not have a recognizable pattern. As evaluation metrics we used *Mean Reciprocal Rank (MRR)* and *coverage*, defined in Section 3. *%impr.* is the improvement

[15]Ken Litkowsky's patterns are available from the TREC website: `http://trec.nist.gov`.

[16]Patterns of correct answers to CLEF QA test data are available from JIRS website: `http://sourceforge.net/projects/jirs/`.

Table 3. MRRs obtained with TREC12 and CLEF04 QA test data

PR system / QA test data	**TREC12**	**%impr.**	**CLEF04**	**%impr.**
FuzzyPR	0.3394	-	0.3726	-
JIRS Distance Model	0.3180	6.73%	0.3721	0.13%
JIRS Simple Model	0.2724	24.60%	0.3771	−1.19%
LucenePR	0.2910	16.63%	0.3399	9.62%

(or worsening) that *FuzzyPR* achieves compared to other PR systems, expressed as a percentage.

Tables 3 and 4 show coverage results (as defined in Section 3) for the three PR systems. The tables show that *FuzzyPR* consistently performs better than LucenePR independently of the number of top-ranked passages consulted when tested with both TREC12 and CLEF04 QA test data. Additionally, MRR obtained by *FuzzyPR* shows an improvement of at least 9.62% and coverage at least 14.47% over LucenePR results.

The comparison of the performance of *FuzzyPR* and the two variations of JIRS shows that *FuzzyPR* performs consistently better in terms of both MRR and coverage, for TREC12 QA test data. Compared to the second best PR system (JIRS *Distance* Model), MRR is improved by 6.73% and coverage by 4.15%. As Table 4(b) shows, *FuzzyPR* tested with CLEF04 QA test data in general (18 out of 20 cases) achieves slightly better coverage than JIRS. Table 4 also reveals that although *FuzzyPR* fails to boost coverage at the ranks 1 to 3, at ranks 4 to 20 it achieves a 0%-7.87% higher coverage than the second best PR system i.e. JIRS *Distance* Model.

However, Table 3 also shows that JIRS *Simple* Model achieves a MRR of 0.3771, which is 1.2% better than *FuzzyPR* in terms of MRR. This fact indicates that sometimes answering passages contained in this collection do not conform well to the *reformulation intuition*. However, this only seems to affect the ability to boost answering passages to higher ranks because JIRS *Simple* Model falls behind JIRS Distance Model and *FuzzyPR* for coverage of passages ranked 4 through 20.

FuzzyPR was tuned using TREC12 QA test data, which might bias the results obtained with the TREC12 corpora. However, Table 4(b) shows that *FuzzyPR* achieves also the highest coverage at ranks 4 to 20 for CLEF04 QA test data. As Gómez-Soriano et al. (2005a) evaluated JIRS with CLEF Spanish, Italian, and French QA test data it is reasonable to assume that JIRS system parameters have been optimized for these languages.

FuzzyPR performs better in general than JIRS due to the incorporation of two additional fuzzy concept besides those included in the JIRS Distance Model: 1) terms are importance-weighted using inverse document frequencies and 2) instead of n-grams the similarity method uses subsequences of n question

Table 4. The PR system coverages tested with (a) TREC12 and (b) CLEF04 data

(a)

	FuzzyPR	LucenePR	JIRS_SM	JIRS_DM
1	0.250	0.224 (11.8%)	0.222 (12.5%)	0.243 (2.7%)
2	0.358	0.305 (17.2%)	0.270 (32.7%)	0.320 (11.8%)
3	0.418	0.350 (19.5%)	0.299 (40.0%)	0.384 (9.1%)
4	0.450	0.371 (21.3%)	0.347 (29.8%)	0.421 (7.0%)
5	0.487	0.403 (20.9%)	0.370 (31.4%)	0.450 (8.2%)
6	0.518	0.424 (22.4%)	0.405 (28.1%)	0.479 (8.3%)
7	0.542	0.434 (24.9%)	0.431 (25.7%)	0.492 (10.2%)
8	0.568	0.453 (25.6%)	0.447 (27.1%)	0.508 (11.9%)
9	0.582	0.479 (21.4%)	0.479 (21.5%)	0.532 (9.4%)
10	0.595	0.495 (20.2%)	0.489 (21.5%)	0.548 (8.6%)
11	0.611	0.505 (20.8%)	0.495 (23.4%)	0.558 (9.4%)
12	0.616	0.524 (17.6%)	0.505 (21.9%)	0.569 (8.3%)
13	0.621	0.529 (17.4%)	0.521 (19.2%)	0.579 (7.2%)
14	0.624	0.537 (16.2%)	0.527 (18.5%)	0.590 (5.7%)
15	0.624	0.547 (13.9%)	0.529 (17.9%)	0.595 (4.8%)
16	0.626	0.550 (13.9%)	0.532 (17.8%)	0.603 (3.8%)
17	0.632	0.558 (13.2%)	0.548 (15.3%)	0.609 (3.8%)
18	0.637	0.561 (13.6%)	0.556 (14.6%)	0.611 (4.2%)
19	0.637	0.561 (13.6%)	0.564 (13.0%)	0.616 (3.3%)
20	0.645	0.563 (14.5%)	0.571 (12.8%)	0.619 (4.2%)

(b)

	FuzzyPR	LucenePR	JIRS_SM	JIRS_DM
1	0.283	0.272 (4.1%)	0.322 (−12.1%)	0.300 (−5.6%)
2	0.378	0.372 (1.5%)	0.389 (−2.9%)	0.372 (1.5%)
3	0.439	0.394 (11.3%)	0.411 (6.8%)	0.444 (−1.2%)
4	0.494	0.422 (17.1%)	0.450 (9.9%)	0.483 (2.3%)
5	0.533	0.439 (21.5%)	0.472 (12.9%)	0.494 (7.9%)
6	0.556	0.456 (21.9%)	0.494 (12.4%)	0.528 (5.3%)
7	0.561	0.472 (18.8%)	0.522 (7.4%)	0.544 (3.1%)
8	0.572	0.472 (21.2%)	0.528 (8.4%)	0.567 (1.0%)
9	0.572	0.483 (18.4%)	0.533 (7.3%)	0.572 (0.0%)
10	0.594	0.489 (21.6%)	0.561 (5.9%)	0.583 (1.9%)
11	0.600	0.489 (22.7%)	0.561 (6.9%)	0.583 (2.9%)
12	0.617	0.489 (26.1%)	0.567 (8.8%)	0.594 (3.8%)
13	0.622	0.489 (27.3%)	0.567 (9.8%)	0.600 (3.7%)
14	0.628	0.500 (25.6%)	0.578 (8.7%)	0.606 (3.7%)
15	0.628	0.506 (24.2%)	0.578 (8.7%)	0.617 (1.8%)
16	0.639	0.506 (26.4%)	0.578 (10.6%)	0.617 (3.6%)
17	0.639	0.506 (26.4%)	0.578 (10.6%)	0.617 (3.6%)
18	0.639	0.517 (23.7%)	0.578 (10.6%)	0.622 (2.7%)
19	0.644	0.522 (23.4%)	0.583 (10.5%)	0.628 (2.6%)
20	0.650	0.533 (21.9%)	0.583 (11.4%)	0.633 (2.6%)

terms, jointly with a proximity method, which yields the highest similarity when the terms are adjacent. Furthermore, compared to JIRS Distance Model, *FuzzyPR* also fuzzifies the following concepts: 3) the definition of passage size and 4) question term occurrences in a passage. The last difference is that *FuzzyPR* computes the proximity of the question terms occurring in a passage rather than relying on n-gram or subsequence matching as JIRS do.

6. *OptFuzzyPR* optimization method

FuzzyPR employs several parameters that have an important effect on its performance. These system parameters were used in equations (17), (18) in Section 4.1 and are listed in Table 5.

Table 5. Description of *FuzzyPR* four system parameters

System parameter	**Description**
ANDness_evidence	The degree of ANDness used for combining the two pieces of evidence[17]characterizing an answering passage.
ANDness_WFoQT	The degree of ANDness used for computing the weighted fraction of question terms occurring in a passage (WFoQT).
w_WFoQT	The importance weight of the weighted fraction of question terms.
w_PoQT	The importance weights of the proximity of question terms (PoQT).

In the experiments reported in Section 4 the parameter values shown in Table 5 were obtained by gradually increasing in small steps these values until we noticed that the PR system performance started to decrease. This approach gave us reasonably good results. However, with the idea of improving the results obtained by FuzzyPR even further, we have optimized the values of *FuzzyPR* system parameters using Genetic Algorithms (GA). GA are one of the most well known optimization techniques and numerous previous research has reported good results on the application of GA in a number of different domains, including information retrieval (Tiedemann, 2005). A detailed description of the theory behind the application of GA can be found in Mitchell (1997).

In this section we briefly describe how GA where used to optimize *FuzzyPR* parameters. To simplify the problem, we focused our efforts on improving the

[17]The weighted fraction of question terms occurring in the passage and the proximity of question terms within the passage.

MRR evaluation metric exclusively. However, it must be remarked that changing the parameter values of *FuzzyPR* to improve MRR may cause in turn a decrease in performance in terms of coverage and vice versa. Our preliminary experiments show that this may be indeed the case. In this situation, finding the set of solutions that optimize both MRR and coverage, involves applying other more sophisticated approaches such as multi-objective evolution (Zitzler and Thiele, 1998). These methods provide Pareto-optimal solutions, by finding the optimal values for a set of parameters that improves one performance measure without making the other perform worse.

6.1. Genetic algorithm

Two basic requirements influenced the design of our optimizing GA. First, an appropriate representation had to be used to model *FuzzyPR* system parameters within the GA. Secondly, because *FuzzyPR* takes approximately 1.5 seconds on average to provide answering passages in response to a question, the GA was required to converge quickly. We achieve partially this goal simply by minimizing the number of iterations required and thus the total execution time.

A *chromosome* consists of four genes, each representing one of *FuzzyPR* system parameters whose values needed to be optimized. The parameters that were optimized are briefly described in Table 5. The domain of all parameters was chosen to be the unit interval.

The GA starts with a population of 25 randomly generated chromosomes holding the genes that represent *FuzzyPR* parameters. No duplicated chromosomes were allowed during generation, as these duplicates represent the same solutions.

Then, the fitness of each chromosome representing an individual solution is computed. The fitness function used in GA is in general problem dependent. As no simple mathematical function can be used in this domain, we employed *FuzzyPR* itself to evaluate the fitness of each individual. The gene values contained in the chromosomes of each individual were employed as the actual parameters of *FuzzyPR*. Then, *FuzzyPR* was executed on the corpora of documents by posing questions, to produce a single MRR score. This value was used to rank the fitness of each individual solution produced by the GA.

At each iteration, the GA generates a new population that is evaluated using a selection mechanism. Selection consisted in choosing 50% of the best fitted chromosomes, as the parents of the next generation of solutions. Additionally, we employed the following simple *crossover strategy*. First, we randomly selected two different chromosomes A and B. For each of the four genes contained in a chromosome, we selected one value from parent A and one from B, each with 50% probability. This strategy was found to perform well in practice.

Finally, we applied *mutation* by re-sampling the individuals generated with the crossover strategy and changing each parameter value to a random one within its domain, with a probability of 2%.

To make the genetic algorithm converge quickly, we used the following three strategies. First, we applied *elitism*, in such a way that the best single configuration from the ith iteration survives to the next $i + 1$th iteration. Secondly, we restricted the domains of the parameters to the unit interval, using a set of discrete values in steps of 0.05, e.g. $0, 0.05, 0.1, ..., 1.0$. Thirdly, as Tiedemann (2005) did, our genetic algorithm keeps in memory chromosomes that performed well during evolution. This is done for performance reasons in order to eliminate duplicate runs.

Finally, as there is no obvious stop condition for the GA within this application domain, we used a predetermined number of iterations to decide when to stop. This same strategy was used by Tiedemann (2005) with good results. Currently, this value is set to 250 iterations in our experiments.

6.2. Evaluation of *OptFuzzyPR*

We compared the performances of the optimized versus the non-optimized versions of *FuzzyPRS*. As the performance metric, we used the Mean Reciprocal Rank (MRR).

In our experiments we used a combination of 360 questions from both CLEF03 and CLEF04 corpora, as training and validation data. From this combined group of questions, we randomly selected 25% of them as training data and the remaining 75% of questions were used as validation data. We used the same approach combining 421 questions of both TREC11 and TREC12 corpora to train and evaluate the performance of *OptFuzzyPR*. The reason for combining questions from these corpora is that the genetic algorithm suffers from overfitting, when the amount of training data is too small.

Fig. 3 show the results of *OptFuzzyPR* with optimized parameters compared to those of *FuzzyPR* using the parameters found in an ad-hoc way.

As Fig. 3 indicates, using the optimized parameters obtained by the GA within *OptFuzzyPR* improves the performance over the non-optimized version of *FuzzyPR* by extra 4% in the case of the CLEF corpora and 2% in the case of the TREC corpora.

Our experiments show that the number of questions in the data set and the number of iterations used in the GA have an important impact on *OptFuzzyPR* performance. The GA suffers from overfitting with a few questions and reaches some local minima with less than 200 iterations. However, each iteration in the GA takes several hours to complete, and therefore, the number of iterations must be kept to the minimum possible.

7. Conclusions and future work

In this paper we presented a novel passage retrieval system: *FuzzyPR*. *FuzzyPR* implements a fuzzy logic based interpretation of the *reformulation intuition*.

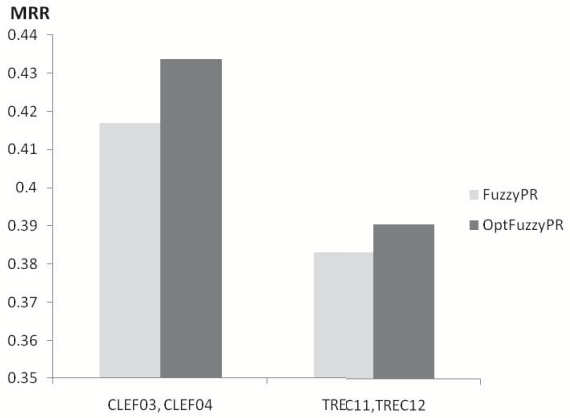

Figure 3. The MRRs of FuzzyPR with and without optimized parameters on CLEF and TREC corpora.

FuzzyPR has three main advantages: 1) its *passage identification and extraction methods* that enable it to retrieve candidate passages from documents at retrieval time thus avoiding the time-consuming indexing process[18] 2) its *language-independence* property, and 3) its ability to handle spelling errors and grammatical inflections.

Our experiments show that *FuzzyPR* achieves a consistently higher MRR and coverage than LucenePR and JIRS on TREC corpora. Furthermore, it performs better in terms of coverage than JIRS on the CLEF corpora at ranks 4 to 20. However, in a few other cases *FuzzyPR* performs slightly worse than JIRS. This seems to indicate that in these cases the answering passages retrieved do not conform well to the reformulation intuition.

The strength of the method employed in *FuzzyPR*, aimed at implementing accurately the reformulation intuition, is also one of its weaknesses, as answering passages that do not conform well to such intuition will degrade performance. For this type of answering passages other approaches based on other models, such as n-gram based systems for example, may perform better.

FuzzyPR has been optimized using GA to achieve the best possible performance. The GA employed in the optimization process was briefly described and a comparison was provided on the performance results obtained by *FuzzyPR* and *OptFuzzyPR*. We found that the optimization process performed on some of the parameters employed in *FuzzyPR* provides additional 4% of improvement on the CLE03 and CLEF04, and 2% on the TREC11 and TREC12 document corpora. However, these small improvements are obtained at the cost of very

[18] A non-optimized method in Java for segmenting and indexing the ACQUAINT corpus took 4 hours on an AMD64 3400+ with 2 GB RAM and RAID 0.

long training and evaluation execution times. The long training times of the GA are due to the repeated execution of *FuzzyPR* over the corpora of documents, as *FuzzyPR* was used in each generation to asses the fitness of each individual solution generated by the GA. The current execution times of our GA-based optimization are in the order of 4-5 days of continued processing. Fortunately, the long training required to optimize *OptFuzzyPR* parameter values needs to be performed only once per each corpus of documents.

As discussed in Section 6, finding the optimal set of parameters to optimize both MRR and coverage involves the application of multi-objective evolutionary optimizations. We plan to perform an extensive set of experiments using these techniques to determine the limits in performance that may be achieved by *OptFuzzyPR*.

Another research direction is the use of data fusion in PR systems. Previous research (Tellex et al., 2003; Christensen and Ortiz-Arroyo, 2007) found that these techniques improve performance. The data fusion techniques employ an ensemble of a diversity of PR systems that rank simultaneously the retrieved passages. Then the ranking of all PR systems is fused to provide a single final ranking of the passages retrieved. However, a disadvantage of these methods is that they require very long execution times, proportional to the number of PR systems employed. One possible solution to tackle this problem is to implement an efficient parallelization of the whole data fusion system. However, another possible approach is to use an ensemble of just two components: *OptFuzzyPR* and a model based on an n-gram PR system such as JIRS. Using this ensemble, a simplified data fusion mechanism will rank higher those passages retrieved by *OptFuzzyPR* that will be detected as conforming well to the reformulation intuition. Contrarily, in cases where the passages are not conforming well, a higher ranking will be given to the n-gram model. This strategy could be implemented using machine learning techniques to identify the passages that conform well to the reformulation intuition.

Finally, we also plan to evaluate *OptFuzzyPR* with CLEF French and Italian corpora to test our system with a broader range of languages. Our experiments show that *OptFuzzyPR* performs generally better with documents written in the English language compared to documents in Spanish. This fact seems to indicate either, that documents in English conform well to the reformulation intuition or that the models embedded in the similarity measure work better for this language.

Acknowledgments

The authors want to thank the anonymous reviewers for their valuable comments that helped us to improve significantly this paper.

References

Beigbeder, M. and Mercier, A. (2005) An information retrieval model using the fuzzy proximity degree of term occurrences. *Proceedings of the 2005 ACM Symposium on Applied Computing*. ACM Press, 1018-1022.

Brill, E., Lin, J., Banko, M., Dumais, S. and Ng., A. (2001) Data-intensive question answering. In: *Proceedings of the Tenth Text REtrieval Conference (TREC 2001), Gaithersburg, Maryland*. Department of Commerce, National Institute of Standards and Technology, 443–462.

Brown, P., Pietra, S.D., Pietra, V.D. and Mercer, R. (1993) The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics* **19**(2), 263–311.

Christensen, H.U. and Ortiz-Arroyo, D. (2007) Applying data fusion methods to passage retrieval in QAS. In: *Proc. of Multiple Classifier Systems, 7th International Workshop*. **LNCS 4472**, Springer, 82–92.

Cui, H., Sun, R., Li, K., Kan, M. and Chua, T. (2005) Question answering passage retrieval using dependency relations. In: *SIGIR'05: Proc. of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM Press.

Damerau, F. (1964) A technique for computer detection and correction of spelling errors. *Communications of the ACM* **7**(3), 171–176.

de Kretser, O. and Moffat, A. (1999a) Effective document presentation with a locality-based similarity heuristic. In: *Proc. of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, California, USA. Australian Computer Science Communications*, **21**, Springer-Verlag and ACM Press, New York, 113–120.

de Kretser, O. and Moffat A. (1999b) Locality-based information retrieval. In: J.F. Roddick, ed., *Proc. of 10th Australasian Database Conference (ADC 99) 18-21 January Auckland, New Zealand. Australian Computer Science Communications* **21**, Springer-Verlag, 177–188.

Freund, Y., Iyer, R., Shapire, R.E. and Singer, Y. (2003) An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research* **4**, 933–969.

Gaizauskas, R., Greenwood, M., Hepple, M. and Roberts, I. (2003) The University of Sheffield's Trec 2003 q&a experiments. In: *Proceedings of the 12th Text Retrieval Conference*. Department of Commerce, National Institute of Standards and Technology.

Gómez-Soriano, J., Montes, M., Gómez, y, Arnal, E. and Rosso, P. (2005a) A passage retrieval system for multilingual question answering. *Proceedings of 8th International Conference of Text, Speech and Dialogue 2005 (TSD'05)*, **LNCS 3658**, Springer-Verlag.

Gómez-Soriano, J., Montes, M., Gómez, Y., Arnal, E., Villasenor-Pineda, L. and Rosso, P. (2005b) Language independent passage re-

trieval for question answering. In: *Proc. of Fourth Mexican International Conference on Artificial Intelligence MICAI 2005.* **LNAI 3789**, Springer-Verlag.

Huang, J., Huang, X. and Wu, L. (2004) Hot-spot passage retrieval in question answering. In: *Digital Libraries: International Collaboration and Cross-Fertilization. 7^{th} International Conference on Asian Digital Libraries, ICADL 2004.* **LNAI 3334**, Springer-Verlag, Berlin Heidelberg.

Kong, K., Luk, R., Ho, K. and Chung, F. (2004) Passage-based retrieval using parameterized fuzzy set operators. *ACM SIGIR Workshop on Mathematical/Formal Methods for Information Retrieval.* ACM Press.

Larsen, H.L. (2003) Efficient andness-directed importance weighted averaging operators. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **11**, 67–82.

Levenshtein, V. (1966) Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physice Doklady*, **10** (8) 707–710.

Lin, D. (1998) An information-theoretical definition of similarity. In: *Proc. of the International Conference on Machine Learning (ICML).* Morgan Kaufmann, 296–304.

Llopis, F., Ferrández, A. and Vicedo, J. Luis (2002) Text segmentation for efficient information retrieval. In: *Proc. of the Third International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2002), Mexico City, Mexico.* **LNCS 2276**, Springer-Verlag, 373–380.

Mitchell, T.M. (1997) *Machine Learning.* McGraw-Hill Companies, Inc.

Monz, C. (2004) Minimal span weighting retrieval for question answering. In: *Proceedings of the ACM-SIGIR 2004 Workshop on Information Retrieval for Question Answering.* ACM Press, 23–30.

Robertson, S.E., Walker, S., Hancock-Beaulieu, M., Gatford, M. and Payne, A. (1995) Okapi at trec-4. In: *Proceedings of the 4th Text Retrieval Conference (TREC-4).* Department of Commerce, National Institute of Standards and Technology.

Saggion, H., Gaizauskas, R., Heple, M., Roberts, I. and Greenwood, M. (2004) Exploring the performance of boolean retrieval strategies for open domain question answering. In: *Proceedings of the SIGIR 2004 Workshop on Information Retrieval for Question Answering.* ACM Press.

Szczepaniak, P. and Gil, M. (2003) Practical evaluation of textual fuzzy similarity as a tool for information retrieval. In: *Proc. of First International Atlantic Web Intelligence Conference, AWIC 2003. Advances in Web Intelligence.* **LNCS 2663**, Springer, 250–257.

Tellex, S., Katz, B., Lin, J., Marton, G. and Fernandes, A. (2003) Quantitative evaluation of passage retrieval algorithms for question answering. In: *SIGIR'03: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.* ACM Press, 41–47.

Terra, E. and Clarke, C. (2005) Comparing query formulation and lexical affinity replacements in passage retrieval. In: *ELECTRA: Methodologies and Evaluation of Lexical Cohesion Techniques in Real-World Applications, ACM-SIGIR Workshop, Salvador, Brazil*. ACM Press.

Tiedemann, J. (2005) Improving passage retrieval in question answering using NLP. In: *Proceedings of the 12th Portuguese Conference on Artificial Intelligence (EPIA)*. **LNAI 3808**, Springer.

Unsunier, N., Amini, M. and Gallinari, P. (2004) Boosting weak ranking functions to enhance passage retrieval for question answering. In: *Information Retrieval for Question Answering Workshop of SIGIR 2004*. ACM Press.

Vilares, J. and Alonso, M. (2004) Dealing with syntactic variation through a locality-based approach. In: *Proc. of the 11th International Conference on String Processing and Information Retrieval, SPIRE 2004*. **LNCS 3246**, Springer, 255–266.

Zitzler, E. and Thiele, L. (1998) Multiobjective optimization using evolutionary algorithms - A comparative case study. In: *Proc. of 5th International Conference on Parallel Problem Solving from Nature, PPSN V*. **LNCS 1498**, Springer, 292–301.